Electronic Knotted Handkerchief

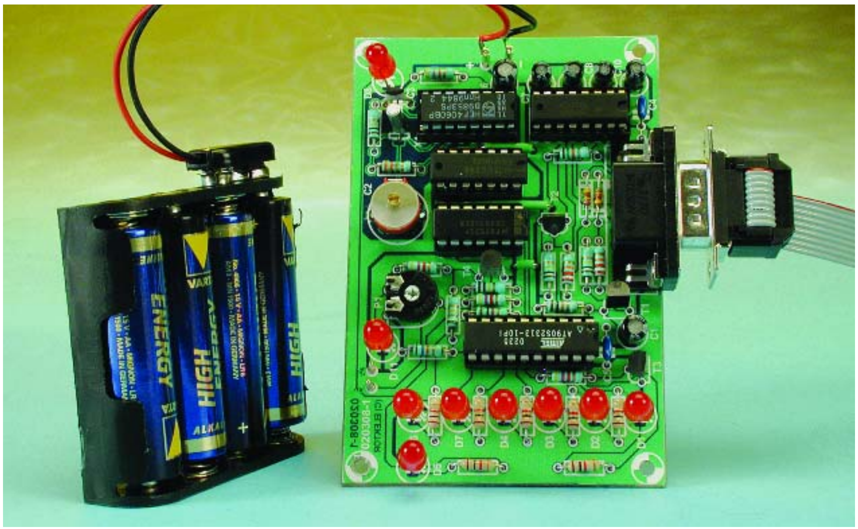
A microcontroller-based notepad

Design by J. Böcker

How many times has the birthday of that rich uncle or that six-monthly appointment at the dentist slipped your mind? How many times have you forgotten to put out the rubbish? The solution is either a little mental training — or the idea presented here: an electronic knotted handkerchief.

Key features

- Space to store from 48 to 94 appointments (depending on the number of distinct events)
- Schedule appointments up to 22 months ahead
- Set times in steps of one hour
- Advance notice period settable from 0 to 255 hours in steps of one hour
- Appointments displayed for a period from 0 to 124 hours (without advance notice), settable in units of four hours
- Battery life: one to two years using four alkaline AA cells (depending on duration and frequency of displayed appointments)



This ‘electronic knotted handkerchief’ is capable of remembering up to six types of appointments. When an appointment approaches, it gives a warning in the form of one of the LEDs (D1 to D4, D6 or D7) flashing. When the appointment is very near, another LED (D5) also flashes. If the capacity for more than six significant events is wanted, then up to 63 different LED patterns can be programmed. Of course, then you will have to keep track of which pattern corresponds to which event, or else your rich uncle will get your tax return and the Inland Revenue will get the flowers!

If no imminent event is indicated, LED D8 flashes every 4 s to show that the device is operating. If the battery is nearly exhausted, D8 blinks once every second.

Programming the device takes place using PC-based software which records the event data and corresponding times and downloads them into the microcontroller. After programming, the application starts up and the connection with the PC can be severed. One disadvantage should be pointed out: when the battery is changed the microcontroller will forget the time, since the clock is implemented in software using a small amount of RAM. The time must therefore be programmed again from the PC.

An end to amnesia

In order to keep the circuit of **Figure 1** as simple as possible, a facility for in-circuit programming has not been provided. The program code (not the appointment data) must be loaded into the processor using an external programming device.

The processor clock is generated using oscillator/divider IC2 along with X1. Here the selection of the particular device is important, as the characteristics vary significantly from manufacturer to manufacturer. If, despite adjustment of C2, the oscillator will not start up, the value of C3 should be reduced slightly.

Since most of the work is done in software, the circuit is relatively easy to construct. The central component is the AVR microcontroller type AT90S2313P from Atmel, which has the following features:

- 20-pin DIL package
- 1K x 16 flash program memory
- 128 bytes RAM
- 128 bytes EEPROM
- 2 timers, one configurable for PWM generation
- integrated comparator
- integrated UART with dedicated baud rate generator

IC2 is not connected directly to the power supply, but rather via R23. This is for the following reason: at low clock frequencies the current consumption of the IC is unreasonably high, probably because during switching both transistors in an output stage briefly conduct at the same time, creating a short circuit. The effect of this on the overall current consumption would be significant if R23 did not limit this current to something under 100 µA. Unfortunately this means that the output signals have much slower rise and fall times, and so they must be squared up using a Schmitt trigger (IC4). The crystal frequency of 32,768 Hz is divided by 2^{13}=8,192 in IC2 and then twice further divided by two in IC3. The result of this is (on pin 1 of IC3) a clock at 1 Hz, which is connected to the INT0 pin of the microcontroller to generate an interrupt signal.

The interrupt service routine sets PD1 high in order to set flip-flop IC3.A. As long as the processor is in the interrupt service routine the low level at the interrupt input is ignored: there is no ‘interrupt within an interrupt’. R16 pulls the set signal to a defined low level when the port output is (for reasons of power consumption) set to a high impedance state.

The anodes of LEDs D1 to D7 are driven via switch T3 from the PWM output PB3/OCI of the microcontroller.

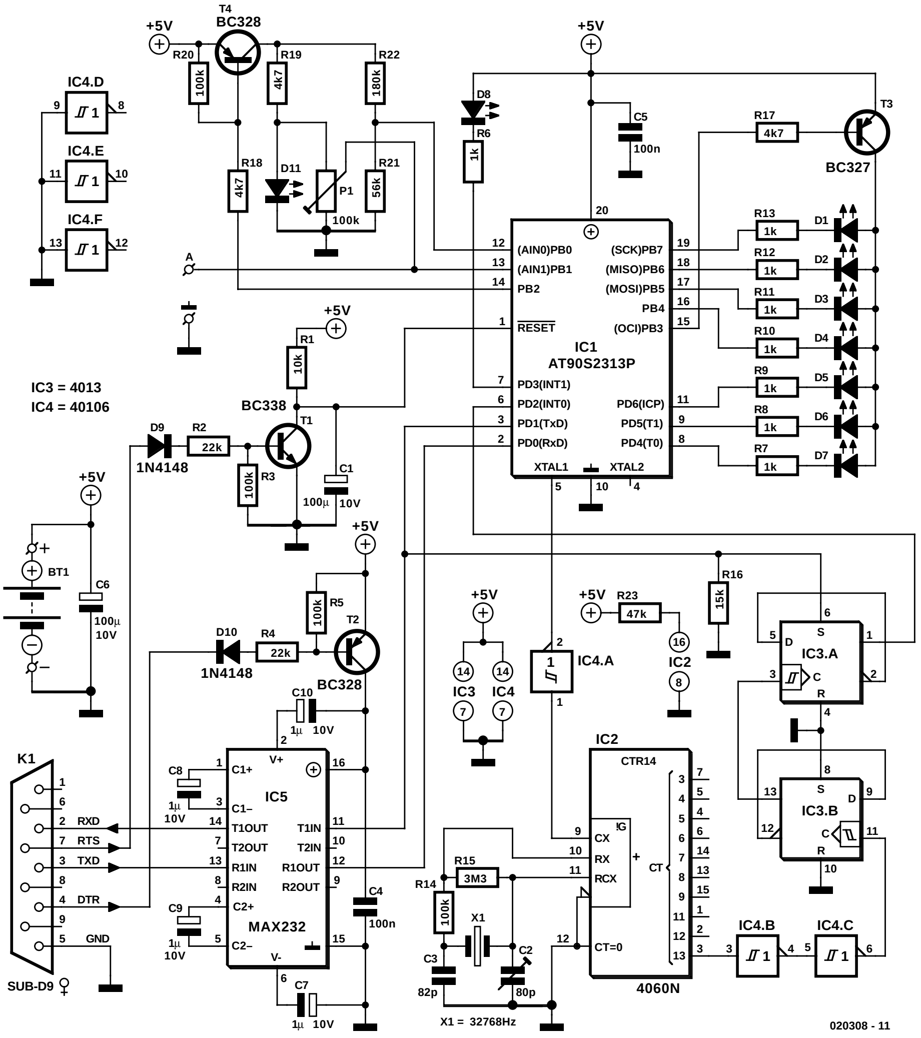


Figure 1. The serial port and battery low detector are switched off when they are not in use.

The battery voltage is measured using the internal comparator via pins PB0 and PB1. This is done just once a day, and even then only when no event (or approaching event) is indicated. If the battery voltage falls below 4 V, a signal appears on port PD7, which is not brought out of the microcontroller.

Lastly we come to the serial interface which uses the well-known MAX232 level shifter (IC5). PD1 drives the TxD signal (also disabling interrupts). Also, the PC can reset the microcontroller using the RTS signal on the serial interface. Without a PC connected T1 is turned off and so a logic high appears at the reset input. If RTS is at logic zero (+12 V), T1 conducts and pulls the reset input low (active). If RTS is at logic one (–12 V), D1 does not conduct and so neither does the transistor. Then C1, which forms part of the power-on reset network with R1, starts to charge and then the program runs.

Power saving

Power-saving measures are essential in a battery-operated circuit that is to run for a year or more. In order to achieve this goal, the LEDs are driven using a PWM signal that can be configured by programming from the PC. This can significantly reduce the current consumption from as much as 4 mA per LED, if the circuit is not to be used in bright ambient lighting. The power-on indicator is driven with a duty cycle of less than 1 % (1/32 s every 4 s); this gives a clearly visible flash.

The FETs in modern components behave – as far as current consumption is concerned – like capacitors. As the operating frequency increases, so does the current consumption. The microcontroller draws around 5 mA in static operation. At a clock frequency in the MHz range this increases by an enormous factor. Here a 32768 Hz watch crystal is used, which is slow, readily available and inexpensive. Also, it is easy to derive a 1 Hz clock by dividing the frequency by 2^{15}.

Since AVR microcontrollers execute practically all instructions in one clock cycle (and are therefore twelve times faster than an 8052), the processor carries out over 30,000 instructions per second. This is more than enough for this application.

Since the baud rate for the microcontroller's integrated UART is derived from the processor clock, communication with the PC is unfortunately only possible at 110 baud. Since the amount of data to be transferred is not enormous, the slow transfer speed does not present a great problem.

Even when the microcontroller draws only 5 mA, that is still too much. Batteries would need to be changed at least once a month, just to keep the microcontroller running; and that doesn't even count the power used by the LEDs and interface chips. An important step in saving power is nevertheless to switch off all unnecessary current-consuming devices, namely the battery voltage measurement circuit (using T4) and the serial interface circuit (using T2).

Of course we cannot switch off the microcontroller in this way, but Atmel does offer several power-saving modes. In sleep mode the microcontroller core is put into a quiescent state, and the device's current consumption falls to 1 mA. However, this is still too much.

In power-down mode, on the other hand, the timers and oscillator are shut down in addition to the processor core. The registers and RAM, however, retain their contents. The microcontroller can only be woken up from this mode by an external interrupt or reset. On interrupt, the processor starts executing code from the address where it was put into the power-saving mode.

Furthermore, the internal components of the CPU (driver stages for the port outputs, the integrated comparator, the timers along with the compare unit used for PWM generation) can be turned off by setting or clearing dedicated control bits in the I/O registers.

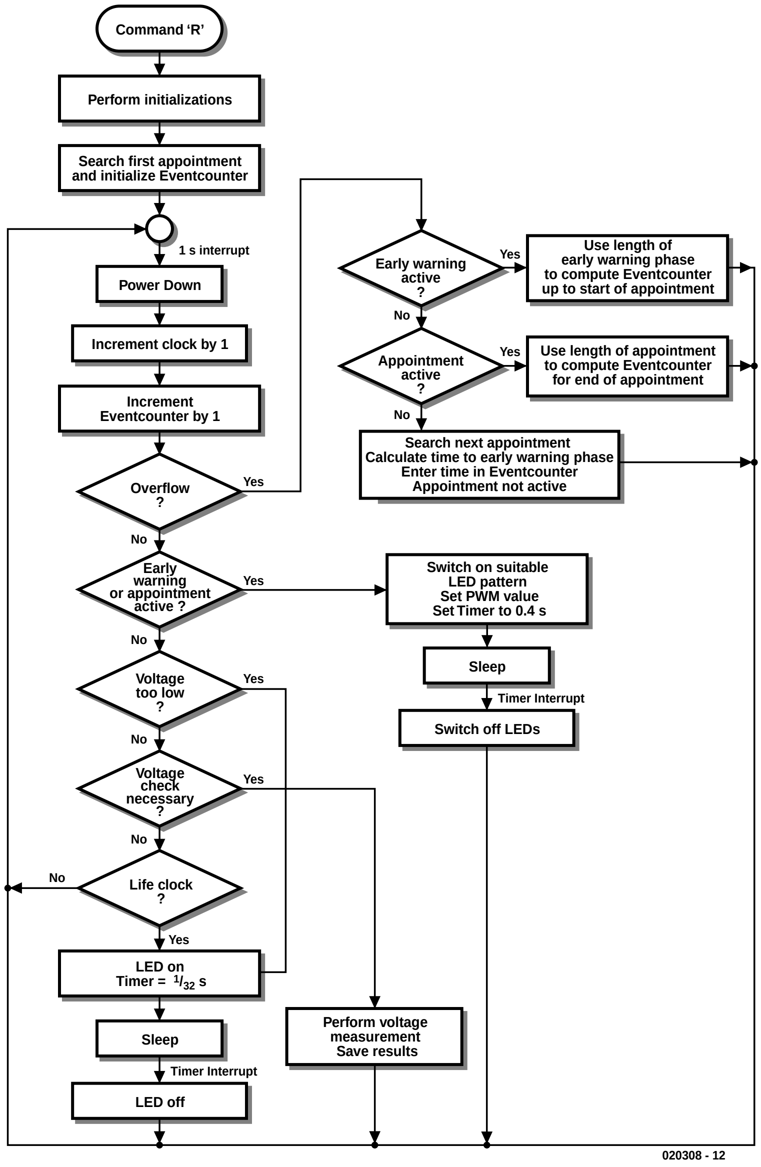

Command 'R'
Perform initializations
Search first appointment and initialize Eventcounter
1 s interrupt
Power Down
Increment clock by 1
Increment Eventcounter by 1
Overflow ?
Yes
No
Early warning or appointment active ?
Voltage too low ?
Voltage check necessary ?
Life clock ?
LED on Timer = 1/32 s
Sleep
Timer Interrupt
LED off
Early warning active ?
Use length of early warning phase to compute Eventcounter up to start of appointment
Appointment active ?
Use length of appointment to compute Eventcounter for end of appointment
Search next appointment Calculate time to early warning phase Enter time in Eventcounter Appointment not active
Switch on suitable LED pattern Set PWM value Set Timer to 0.4 s
Switch off LEDs
Perform voltage measurement Save results
020308 - 12

Figure 2. Flowchart for the operating state.

Software: part 1

The microcontroller program, which is written entirely in assembler code, operates in two different states: the programming/configuration state and the operating state. In the programming state the microcontroller communicates with the PC over the serial interface.

Immediately after a reset the microcontroller emits the sequence of characters 'HELLO' and then waits for a command to arrive. As an extra check that reset has occurred and that the microcontroller is operating correctly, all the LEDs flash.

The available commands which can be sent to the microcontroller are listed in **Table 1**. A few of these commands are for development and debugging; the majority are either for programming events or for testing the hardware.

If the 'R' command is sent, all the LEDs flash once and the controller switches into operating state.

Software: part 2

The operating state is configured to use an absolute minimum of power. The serial interface is no longer checked. Once a second the microcontroller is woken up from the power-down mode by an interrupt, does whatever work is necessary and then returns as quickly as possible to the power-down mode.

Essentially, registers R0 to R3 are used to count seconds from a virtual 'time zero'. All event times are reckoned relative to this zero reference point. When the PC puts the unit into operating mode it stores the time difference between this reference point and the actual time in registers R0 to R3.

Registers R4 to R7 are used to store the number of seconds until the next appointment, or until some other state change occurs, such as at the beginning of the advance warn-

Table I. Command/function overview

Command	Function
s	Enter idle mode
p	Enter power-down mode
e	Set LED pattern brightness to full
a	Turn all LEDs off
N	Turn serial interface off
R	Start program
B	Display flashing LED pattern, Values on stack are: LED pattern; PWM setting; on period in units of 128 s
C	Turn supply to battery low detector on
c	Turn supply to battery low detector off
m	Run continuous battery low detector test and output results continuously over serial interface
v	Push next value on stack
d	Dump virtual memory (EEPROM and RAM)
w	Write data stream into virtual memory. Values on stack are: count; start address

ing period for an event. On each one-second interrupt the value in R4 to R7 is decremented by one. When the value reaches zero, the appropriate action to be taken must be determined. The corresponding state is set and relevant program code segment is executed.

Immediately on entering the operational state, or when an appointment expires, data for the appointment is read from bit-addressable memory space and loaded into registers R4 to R10, R13 and R14.

Registers R11 and R12 form a counter used in the voltage measurement process. This counter is initialised with the hexadecimal value FEAD and incremented every 256 seconds. When the counter rolls over (roughly every 24 hours) a voltage measurement is carried out.

Once the last appointment has passed interrupts are disabled and the unit goes into the power-down mode permanently. In this case the batteries should be removed, since the oscillator is still running and consuming energy.

The flowchart in **Figure 2** shows the general sequence of operations in the operating state.

Software: part 3

The electronic knotted handkerchief is programmed using a PC. This of course requires further software, this time running on the PC. The type of event and the time are recorded, the information is downloaded into the device and the timer started. It is also possible to carry out hardware tests and configure various aspects of the circuit.

The program was written in Java using the free personal version of Borland JBuilder. Although Java might not seem the most obvious choice for this application, it is in fact an eminently suitable language. Unfortunately the program takes rather a long time to start up, and the 'Swing' class responsible for the user interface will run sluggishly on older PCs. The Java Communication API (CommAPI), which can be downloaded from Sun's website, is used to drive the serial port.

It is not possible to just run a Java program directly on a PC. First version 1.3 or 1.4.1 of the Java Run-time Environment (JRE) from Sun must be installed. This is available for free download at
http://java.sun.com/getjava/download.html.

Older versions will not work with this program and should not be used. The Microsoft Java VM supplied with older versions of Windows will not work either.

The program ELTAKO.EXE and the DLL for the CommAPI win32com.dll can be copied to any desired directory (although both files should go in the same directory). In order to install the CommAPI class, the file javax.comm.properties should be copied into the directory JRE\lib, or into whatever directory the RDK was installed. When you open a DOS window and enter the command

```
java -verbose | more
```

you should see the correct path to the RDK. If this does not work, you can always search for the JRE directory.

The program is started by clicking on ELTAKO.EXE.

When the button labelled 'NEW' is clicked, a new event is created in the left-hand area. You can use 'DEL' to delete an event and all the appointments associated with it. When you click on an event, it is displayed in the middle of the window. Here you can now enter the name of the event, the LED pattern which is to correspond to it, the amount of advance notice required and the duration. The duration must be a multiple of 4 hours. In the right-hand area the times for the event are shown. These must be in chronological order. To delete an appointment, the date can be blanked using the backspace key. This may create gaps in the appointment list: press the 'TIDY UP' button, and the times are sorted and deleted entries removed.

Once all the appointments have been entered, they can be saved using the 'SAVE' menu item; you can reload them at a later date using 'LOAD'. Next, select the 'DOWNLOAD' menu item. A new window appears and the data are converted into bit-addressable format. Error messages, along with details about their cause, will appear at the top of the window. Errors should be corrected in the main window before attempting to download again.

If the status line shows 'Data translation OK', you can proceed to connect the device to the PC and press the 'TRANSMIT' button. After a reset (the LEDs flash) the data are transferred and the time counter is set to the correct value. Then the unit is switched to its operating state; it can now be disconnected from the PC. Make sure that the PC's clock is set correctly.

Construction and testing

The circuit board shown in **Figure 3** is rather sparsely populated, and so construction should present no difficulties. The four wire links (one is under the microcontroller) must of course not be overlooked. For this reason the microcontroller should be fitted in a socket. All the other ICs may be fitted in sockets, although this is not compulsory.

The power supply takes the form of four 1.5 V cells, which are most conveniently fitted in the battery compartment of a suitable enclosure. Alternatively a suitable battery holder can be glued into the enclosure.

COMPONENTS LIST

Resistors:
R1 = 10kΩ
R2,R4 = 22kΩ
R3,R5,R14,R20 = 100kΩ
R6-R13 = 1kΩ
R15 = 3MΩ3
R16 = 15kΩ
R17,R18,R19 = 4kΩ7
R21 = 56kΩ
R22 = 180kΩ
R23 = 47kΩ
P1 = 100kΩ preset

Capacitors:
C1,C6 = 100µF 10V radial
C2 = 80pF trimmer
C3 = 82pF
C4,C5 = 100nF
C7-C10 = 1µF 10V radial

Semiconductors:
D1-D8,D11 = LED, red, low current
D9,D10 = 1N4148
IC1 = AT90S2313-10PC, programmed, order code **020308-41**
IC2 = 4060N
IC3 = 4013
IC4 = 40106
IC5 = MAX232
T1 = BC338
T2,T4 = BC328
T3 = BC327

Miscellaneous:
BT1 = 4 AA or AAA batteries with holder
K1 = 9-way sub-D socket (female), PCB mount, angled pins
X1 = 32.768kHz quartz crystal
Disk, order code **020308-11** or Free Download

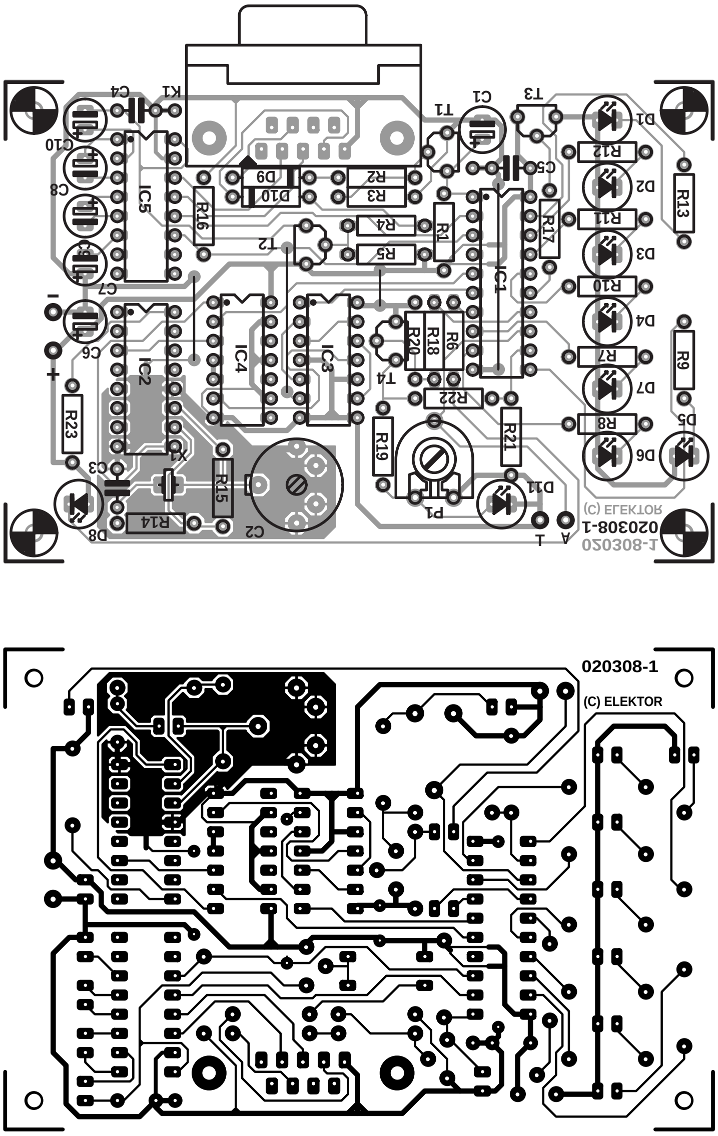


Figure 3. Everything fits on one printed circuit board.

First IC2 and IC4 should be fitted in their sockets and the device switched on. C2 should be adjusted until the oscillator operates reliably. The current consumption of the circuit should be no higher than 30 µA with these components fitted.

Before fitting the microcontroller into its socket it should be programmed from the file MAIN.HEX. You can do this yourself (the software is available from the *Elektor Electronics* website or on diskette number **020308-11**) or a ready-programmed microcontroller can be obtained (product number **020308-41**).

Before fitting the unit into an enclosure it is worth going through the hardware test and configuration procedure. For this, the electronic knotted handkerchief should be connected to the serial port of a PC using a straight-through RS232 cable with a male connector at one end and a female connector at the other. The program HW-TEST should then be run. Press the RESET button and a connection will be established. You can now test operation by turning all the LEDs on and off individually. The slider allows you to set the LED brightness, and when 'OK' is clicked the value is stored permanently.

The final step is to calibrate the battery low detector. First activate the circuit using the 'POWER ON' button and then adjust P1 for a voltage of 0.9 V between test point A and ground. Once the tests have been completed the connection can be broken using the 'UNPLUG' button.

The timing accuracy of the unit leaves a little to be desired. This is because normal crystals have a relatively large tolerance relative to their nominal frequency, even though according to the data sheet the maximum error in the crystal frequency should be only 10 ppm. In practice errors of more than 3 s per month are found.

(020308-1)

Free Downloads

- PC and microcontroller software.
  File number: 020308-11.zip
- PCB layout in PDF format.
  File number: 020308-1.zip

www.elektor-electronics.co.uk/dl/dl.htm, select month of publication.